Own your ClickHouse data with

Altinity.Cloud® Anywhere

Robert Hodges and Alexander Zaitsev

Let's make some introductions

Us

Database geeks with decades of experience in DBMS and applications

You

App developers looking to build real-time analytics to solve business problems*

* You also like Kubernetes ;)

ClickHouse support and services including Altinity.Cloud®
Authors of Altinity Kubernetes Operator for ClickHouse
and other open source projects

What's Altinity.Cloud Anywhere?

ClickHouse is a real-time analytic database

Understands SQL

Runs on bare metal to cloud

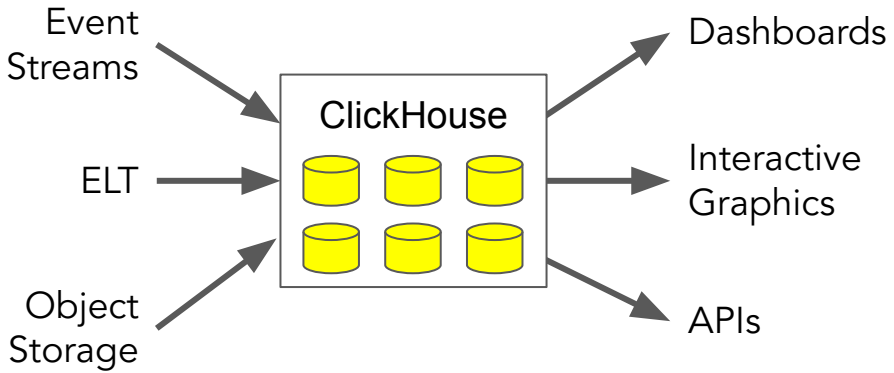
Shared nothing architecture

Stores data in columns

Parallel and vectorized execution

Scales to many petabytes

Is Open source (Apache 2.0)



It's the core engine for low-latency analytics

Altinity.Cloud is a zero-maintenance SaaS for ClickHouse

Complete automation of operations with baked in DBA support

Cost and performance optimized for real-time analytics

Supports all versions and features of ClickHouse (also experimental ones!)

Run in any AWS/GCP region or your own Kubernetes clusters

Pure SaaS is convenient, but many users need more choices to meet business requirements

Run in your own VPCs for security compliance

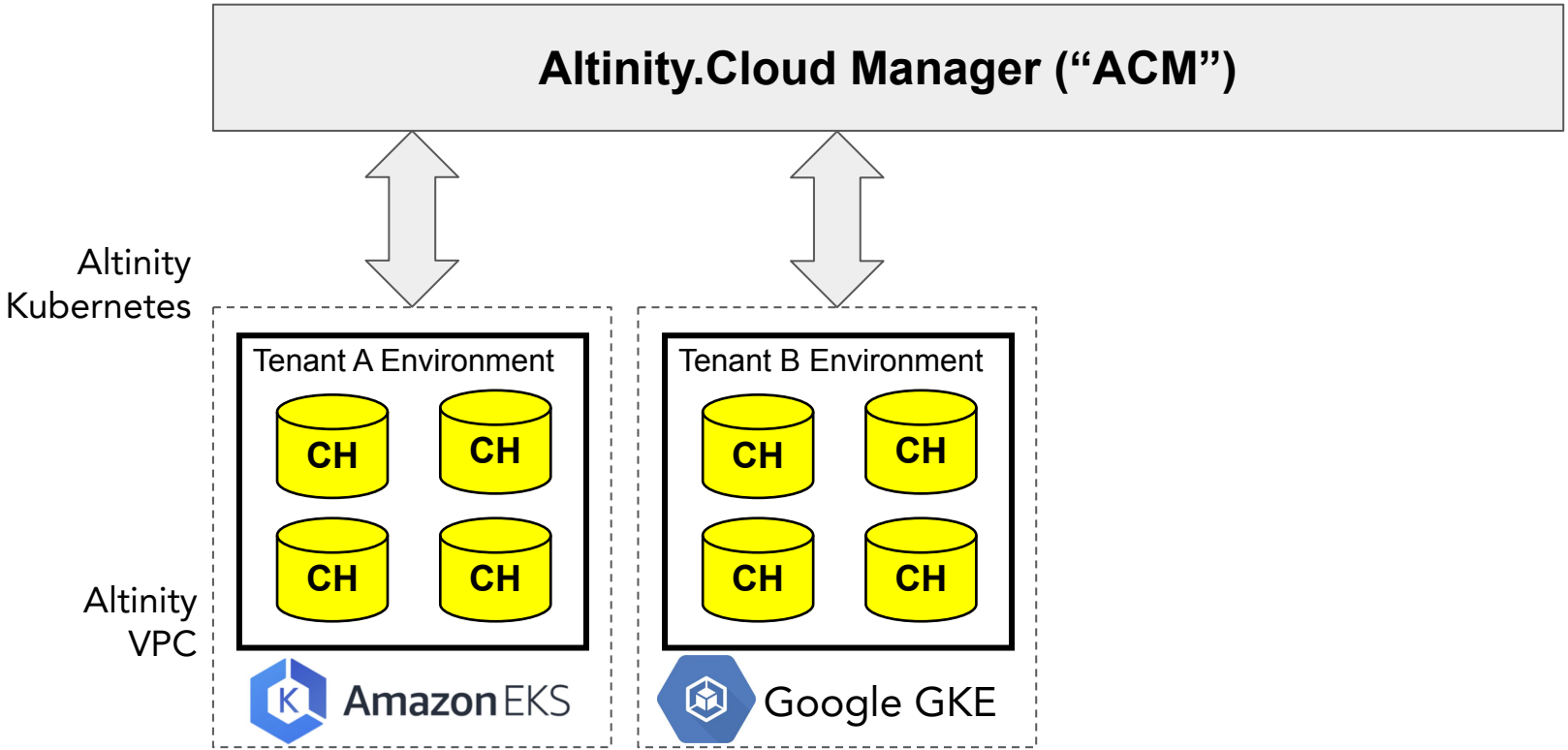
Use your own infrastructure to control costs

Keep analytics close to on-prem data sources

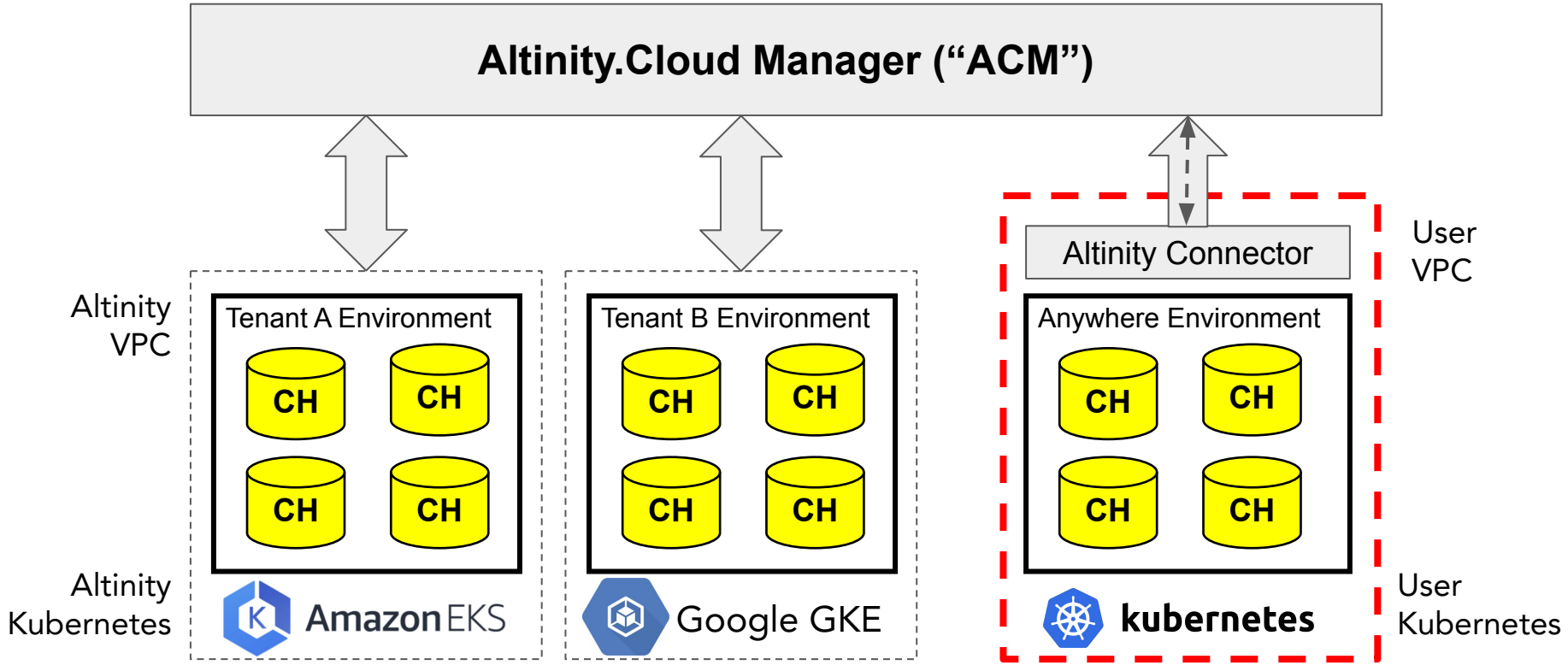
Use 100% open source to avoid vendor lock-in

DEMO TIME!

Altinity.Cloud runs ClickHouse in Kubernetes "environments"



Altinity.Cloud Anywhere can run in your Kubernetes



Getting up and running with Altinity.Cloud Anywhere

Preparing your Kubernetes

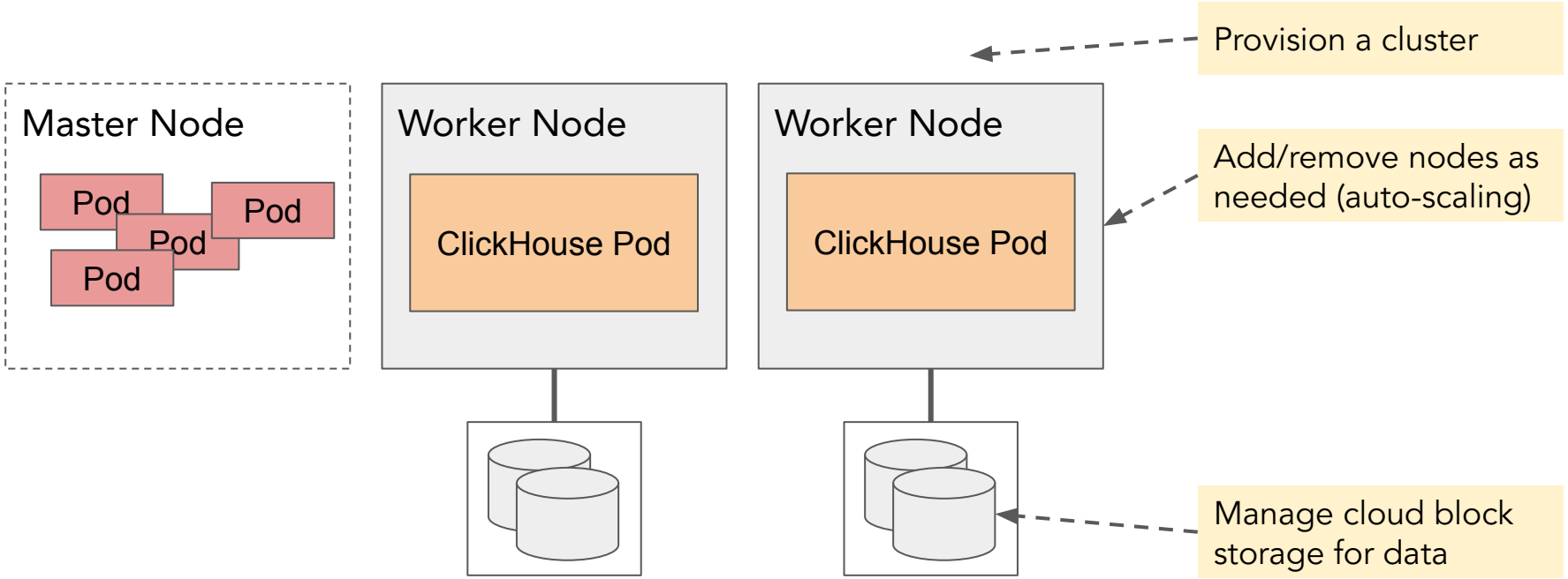
Choosing a Kubernetes distribution

Currently certified distributions

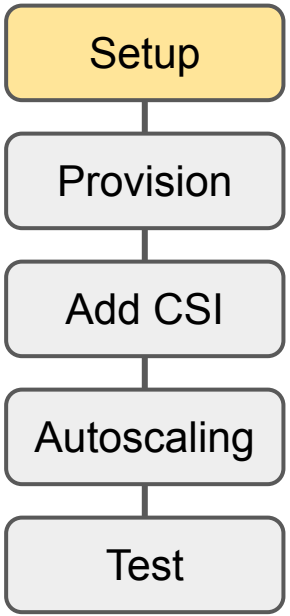
* Used for demo/test only

Pending certifications

Your Kubernetes setup needs to get a few things right

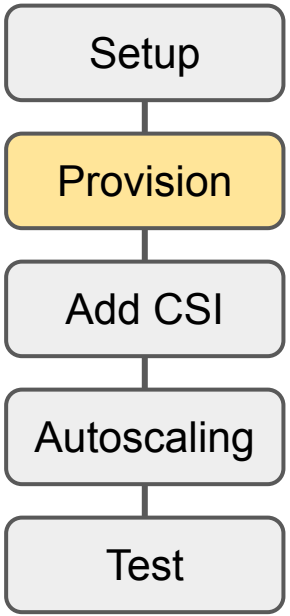


Setup to provision EKS Cluster using eksctl



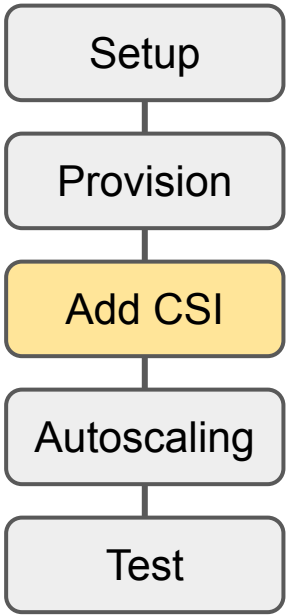
1. Bring up jump VM in AWS
2. Install AWS CLI
3. Install eksctl
4. Install kubectl (1.24 or greater)
5. Install aws-iam-authenticator
6. Add AWS IAM permissions to VM to create EKS clusters

Provision Kubernetes cluster on EKS



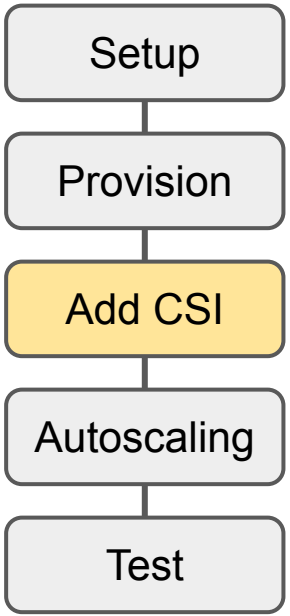

```
eksctl create cluster -f - << EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: ${CLUSTER_NAME}
  region: ${AWS_DEFAULT_REGION}
  version: "1.23"
  tags:
    karpenter.sh/discovery: ${CLUSTER_NAME}
managedNodeGroups:
  - instanceType: m5.large
    amiFamily: AmazonLinux2
    name: ${CLUSTER_NAME}-ng
    desiredCapacity: 2
    minSize: 1
    maxSize: 10
iam:
  withOIDC: true
EOF
```

Add CSI driver to managed storage (Kubernetes 1.23+)



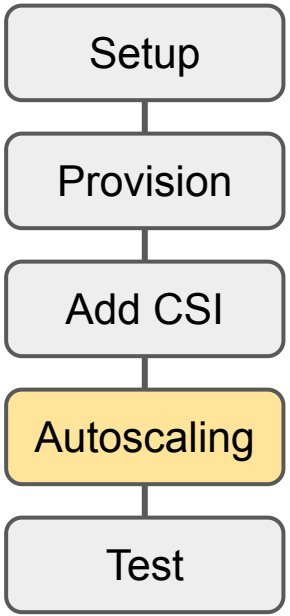

```
eksctl create iamserviceaccount \
  --name ebs-csi-controller-sa \
  --namespace kube-system \
  --cluster ubuntu-altinity-cloud-anywhere-demo \
  --attach-policy-arn
arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \
  --approve \
  --role-only \
  --role-name AmazonEKS_EBS_CSI_DriverRole

eksctl create addon --name aws-ebs-csi-driver \
  --cluster ${CLUSTER_NAME} \
  --service-account-role-arn
arn:aws:iam::${AWS_ACCOUNT_ID}:role/AmazonEKS_EBS_CSI_DriverRole \
  --force
```

Add matching storage class(es) for CSI driver



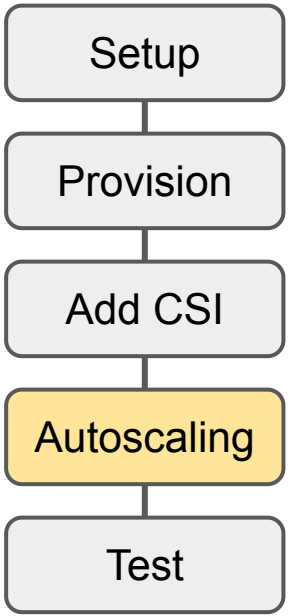

```
kubectl apply -f - <<EOF
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: gp2
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: ebs.csi.aws.com
parameters:
  fsType: ext4
  type: gp2
reclaimPolicy: Delete
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
EOF
```

Add Karpenter to allocate/deallocate VMs automatically



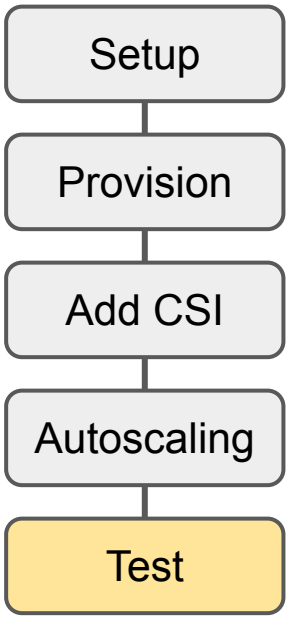
1. Install Helm
2. Install Karpenter using getting started with eksctl instructions
   a. Create Karpenter infra and IAM roles
   b. Create IAM identity mapping so new VMs can join cluster
   c. Create IAM role for KarpenterController so it can launch VMs
   d. Install Karpenter Helm Chart
   e. Create a Karpenter provisioner to manage VMs

Here's what a Karpenter provisioner looks like




```
cat <<EOF | kubectl apply -f -
apiVersion: karpenter.sh/v1alpha5
kind: Provisioner
metadata:
  name: default
spec:
  limits:
    resources:
      cpu: 1000
  providerRef:
    name: default
  ttlSecondsAfterEmpty: 30
---
apiVersion: karpenter.k8s.aws/v1alpha1
kind: AWSNodeTemplate
metadata:
  name: default
spec:
  subnetSelector:
    karpenter.sh/discovery: ${CLUSTER_NAME}
  securityGroupSelector:
    karpenter.sh/discovery: ${CLUSTER_NAME}
EOF
```

Try it out!



Make sure your Kubernetes cluster can provision nodes and allocate storage!

Try installing a simple application like nginx and prove that

1. Karpenter can allocate a VM for it.
2. Storage allocation on block storage works.

Common problems:

- You missed IAM privileges somewhere
- CSI driver and matching storageclass missing/invalid
- Karpenter provisioner not correctly defined

Seems complicated! Isn’t there a better way?

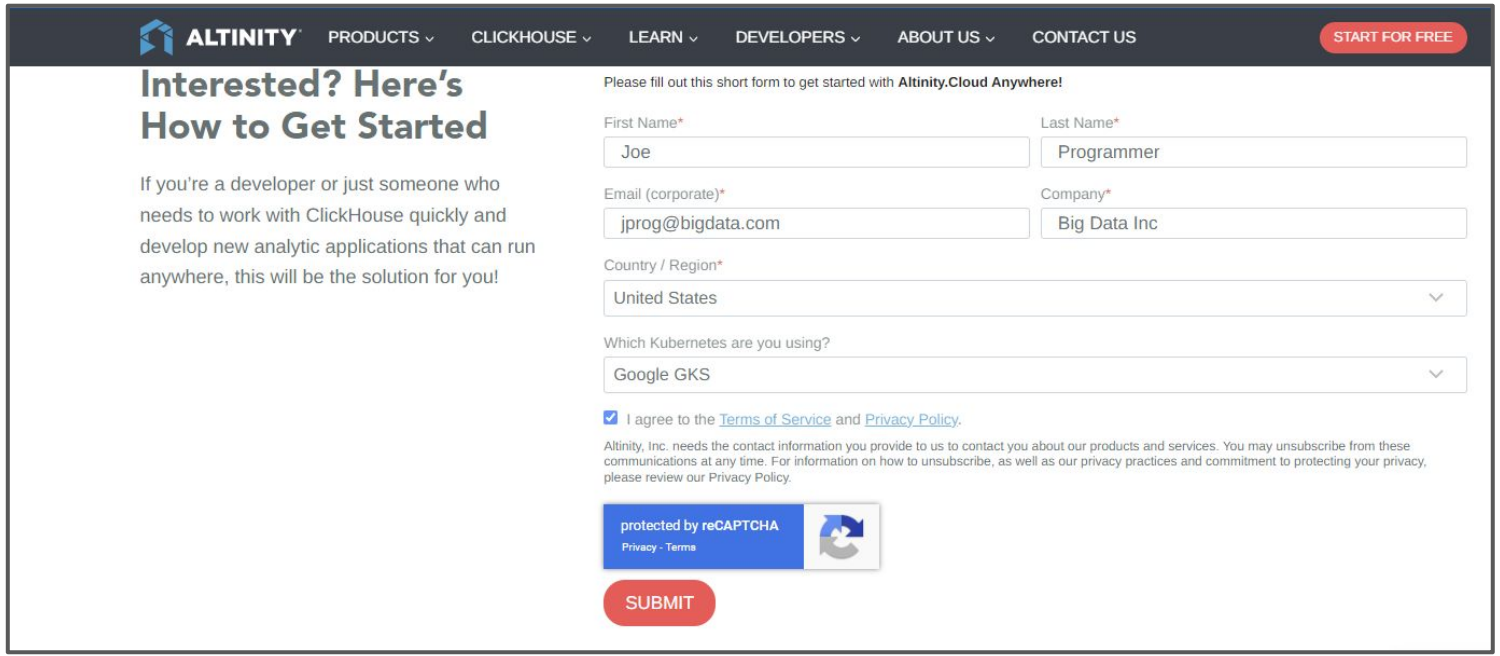
Yes!!

Starting in Q1 2023, Altinity.Cloud will offer full provisioning of EKS from soup-to-nuts.

All you need to supply is an IAM role and a VPC.

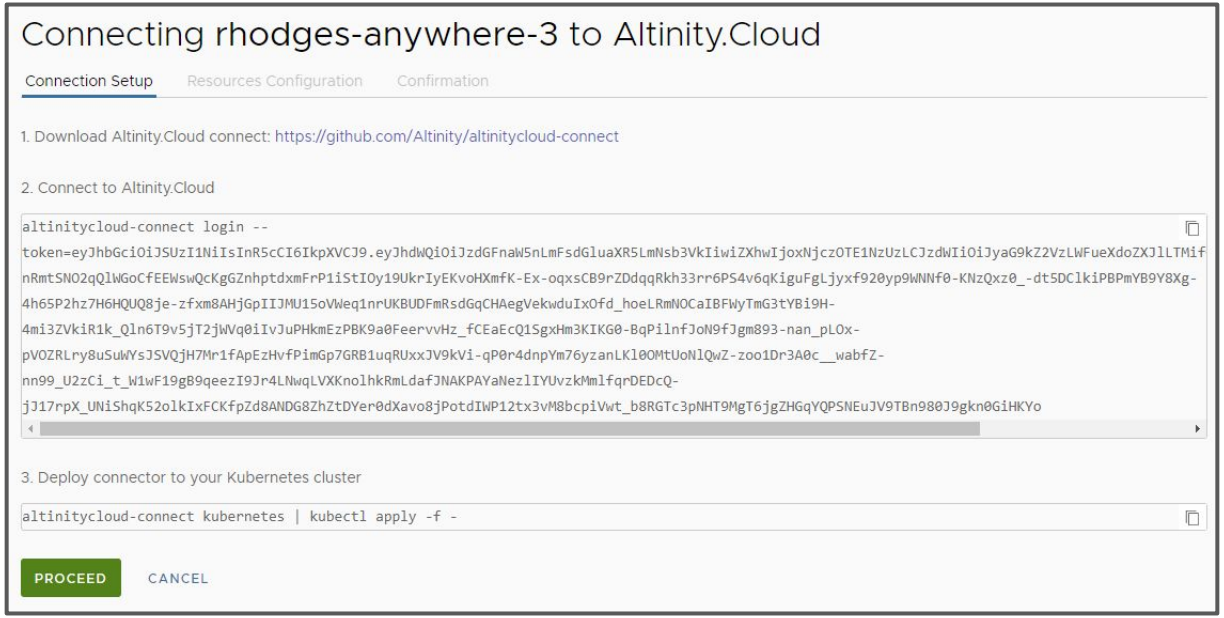
Connecting to Altinity.Cloud

Get an Altinity.Cloud account (if you don't have one)



https://altinity.com/altinity-cloud-anywhere

Sign into Altinity.Cloud and find your environment



https://acm.altinity.cloud

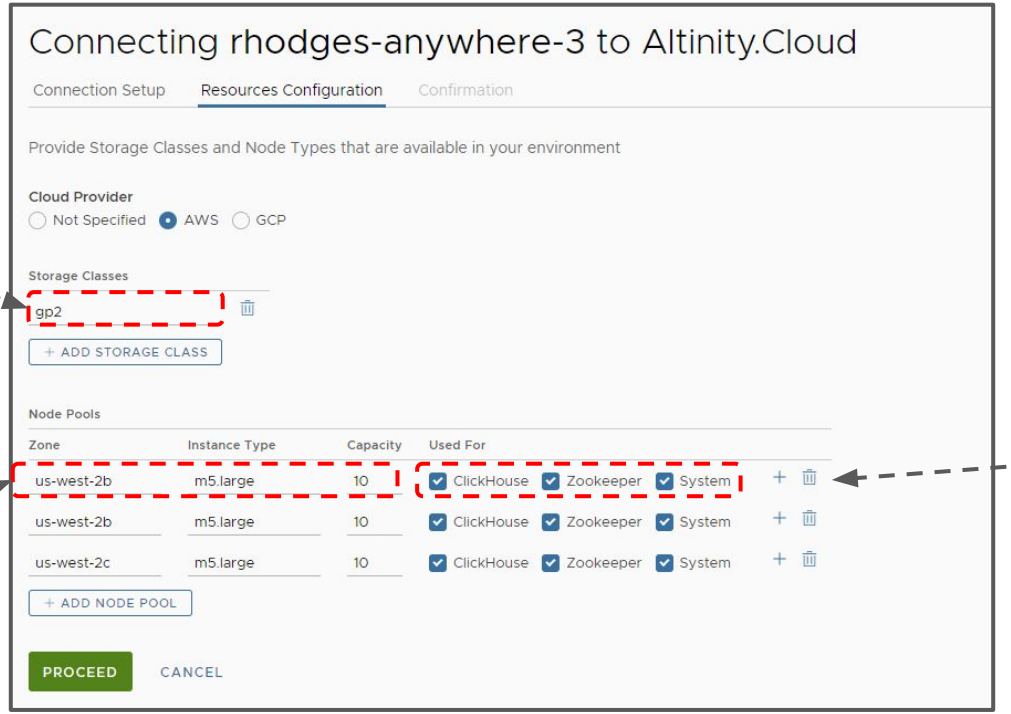
Install Altinity Connector and set up connection

```
# Download altinitycloud-connect.
curl -sSL
https://github.com/altinity/altinitycloud-connect/releases/download/v0.9.3/al
tinitycloud-connect-0.9.3-linux-amd64 -o altinitycloud-connect \
&& chmod a+x altinitycloud-connect \
&& sudo mv altinitycloud-connect /usr/local/bin/

# Login to Altinity.Cloud.
altinitycloud-connect login --token=<registration token>

# Pipe setup commands to deploy connector to your Kubernetes cluster.
altinitycloud-connect kubernetes | kubectl apply -f -
```

Fill out resource configuration

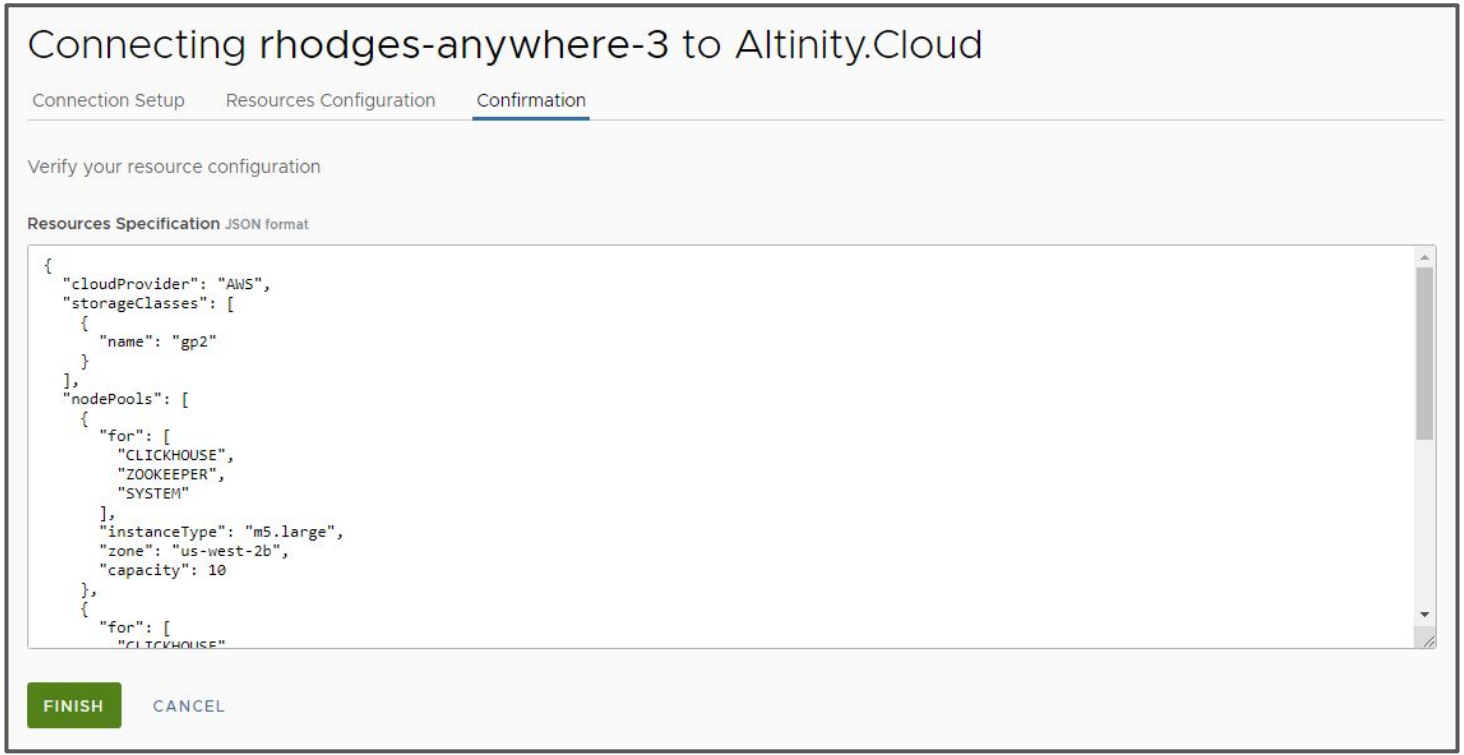
Block storage types

Node pool AZs and machine types



Services that will use the node pool

Review and Press FINISH to initiate setup

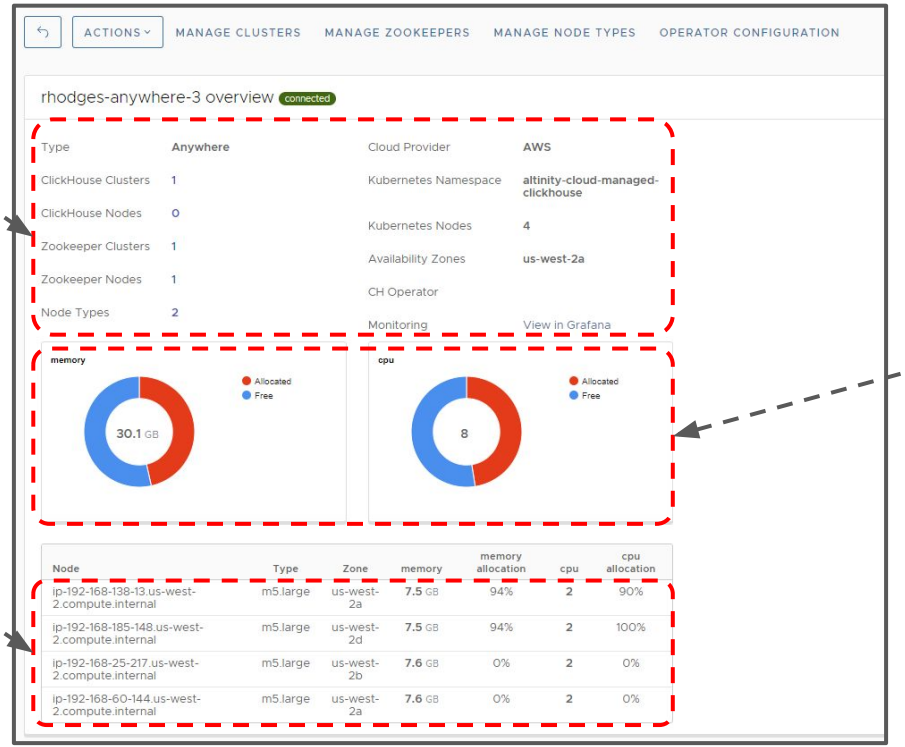


Altinity.Cloud Anywhere environment dashboard

Environment statistics

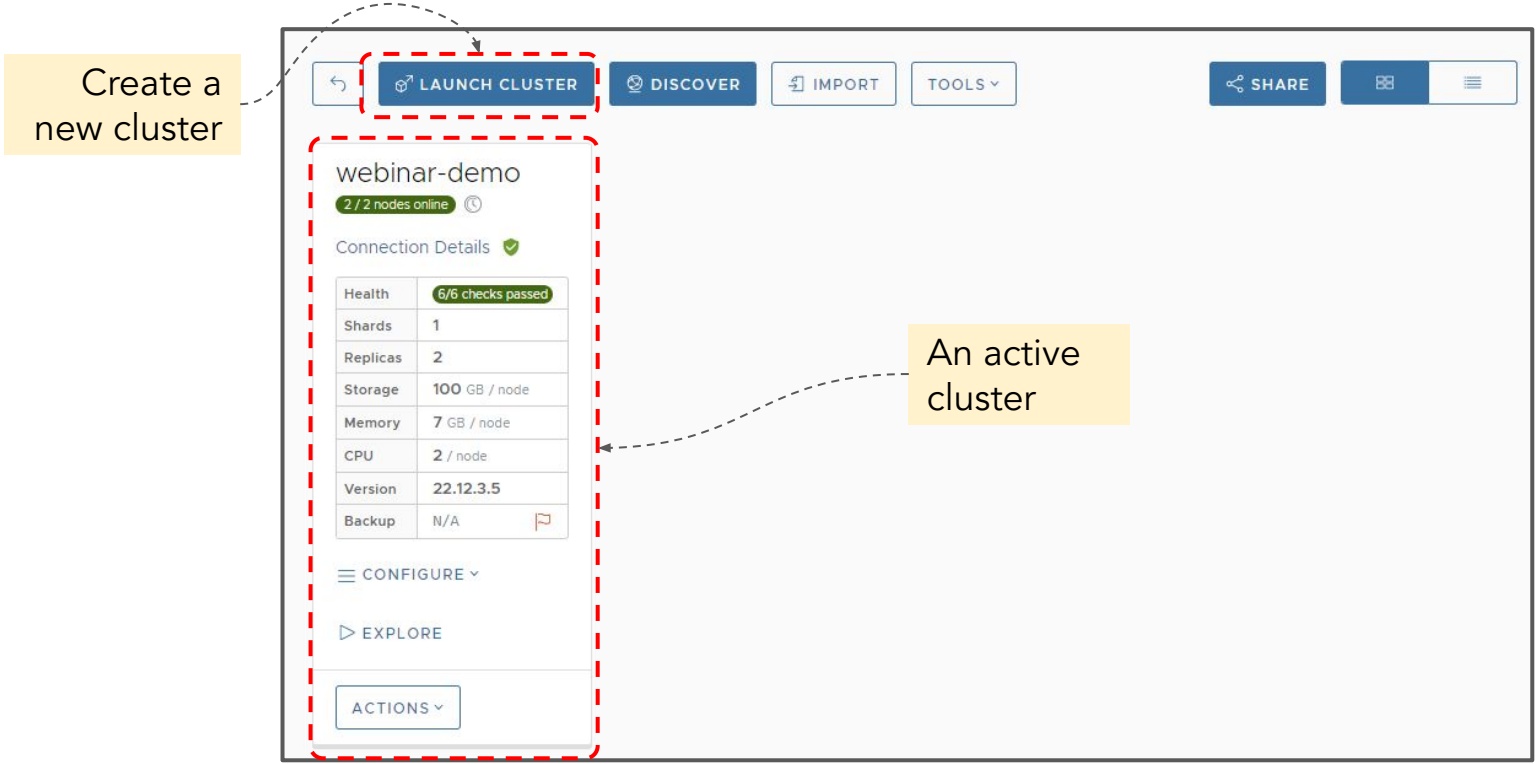
Current nodes

Allocated RAM and storage

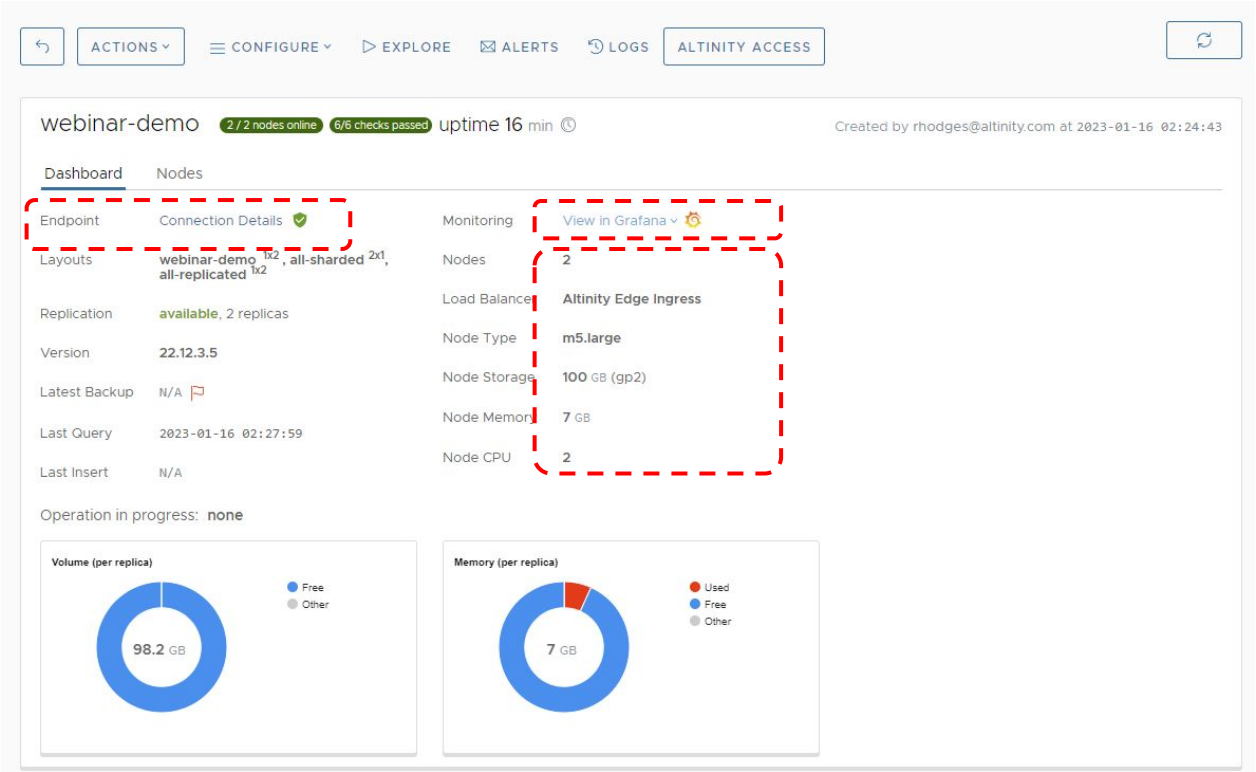


Working with Clusters in Anywhere

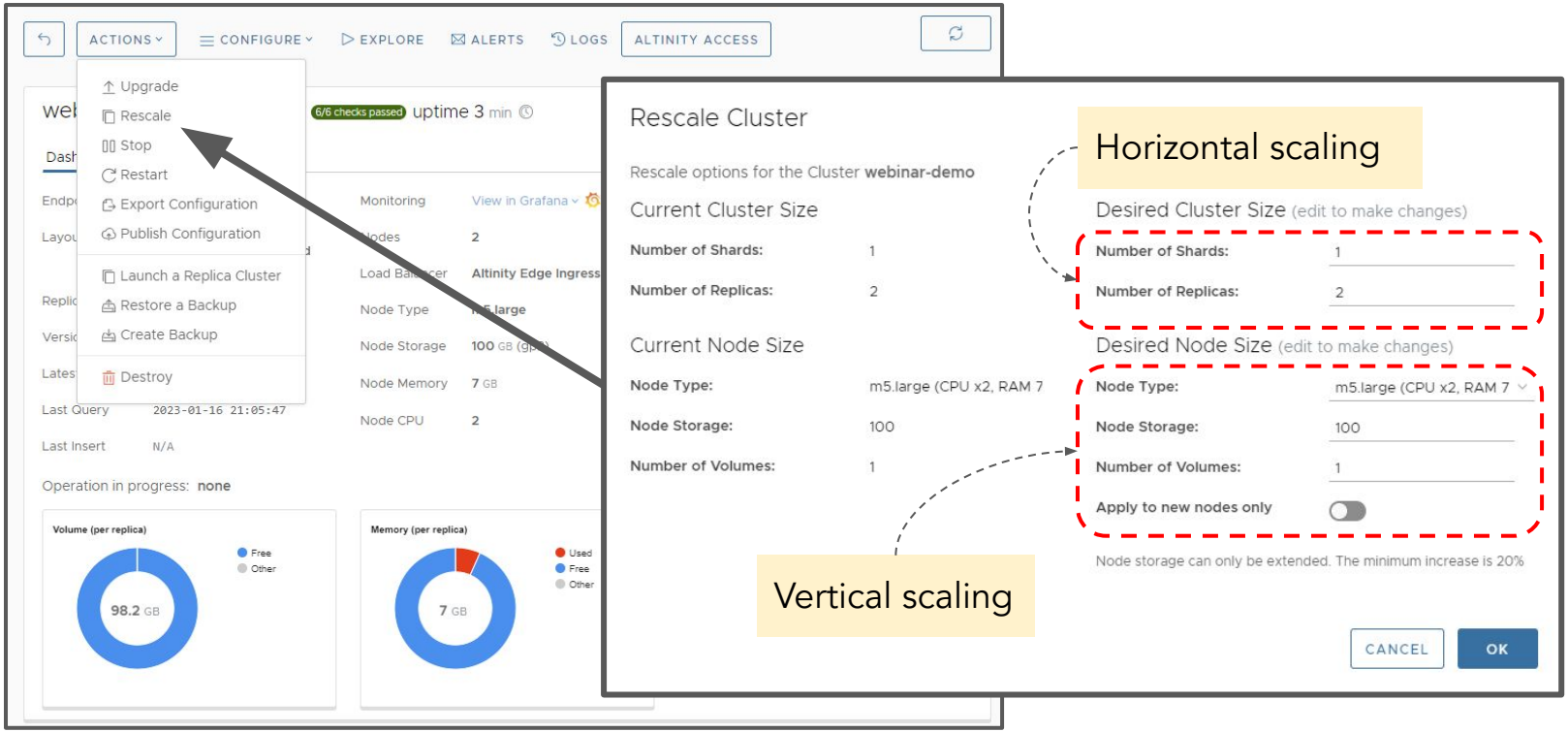
Cluster view is the same as Altinity.Cloud



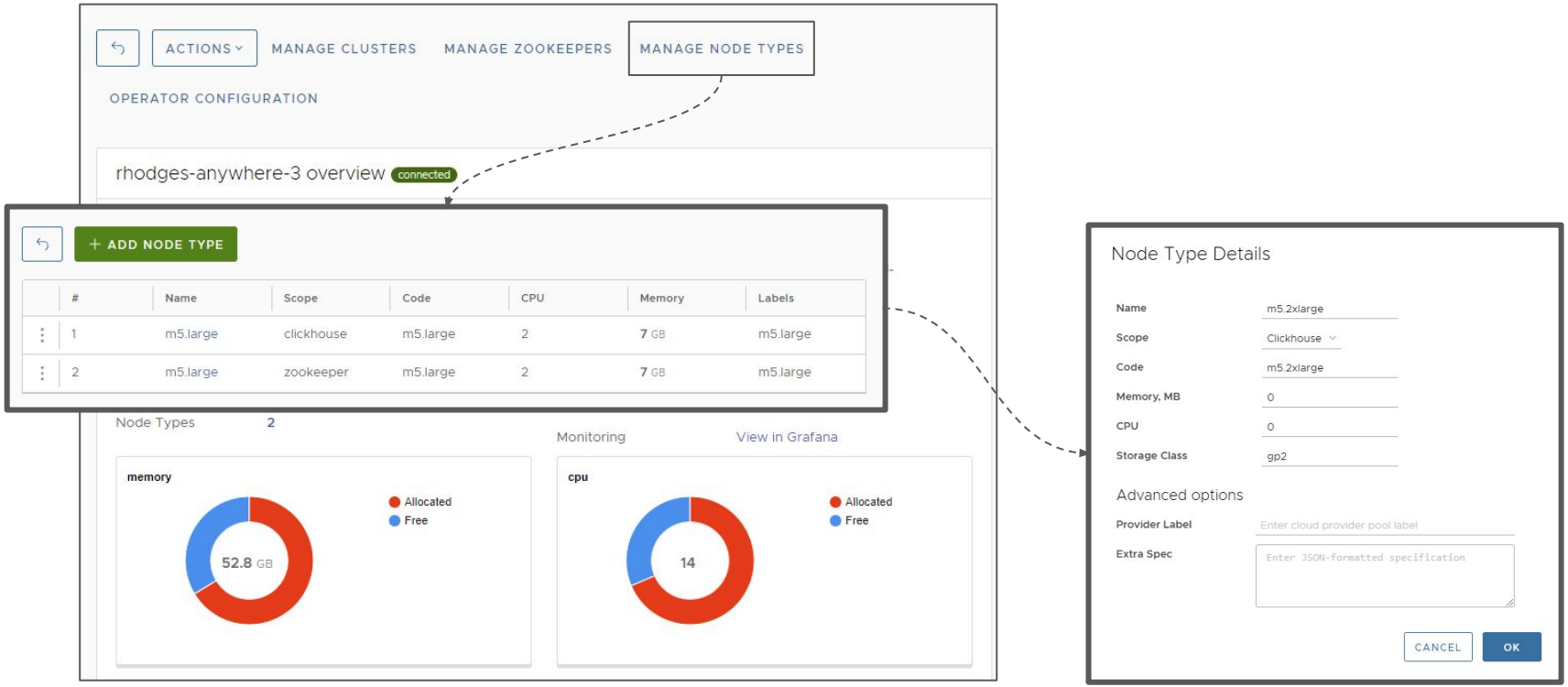
So is the cluster dashboard



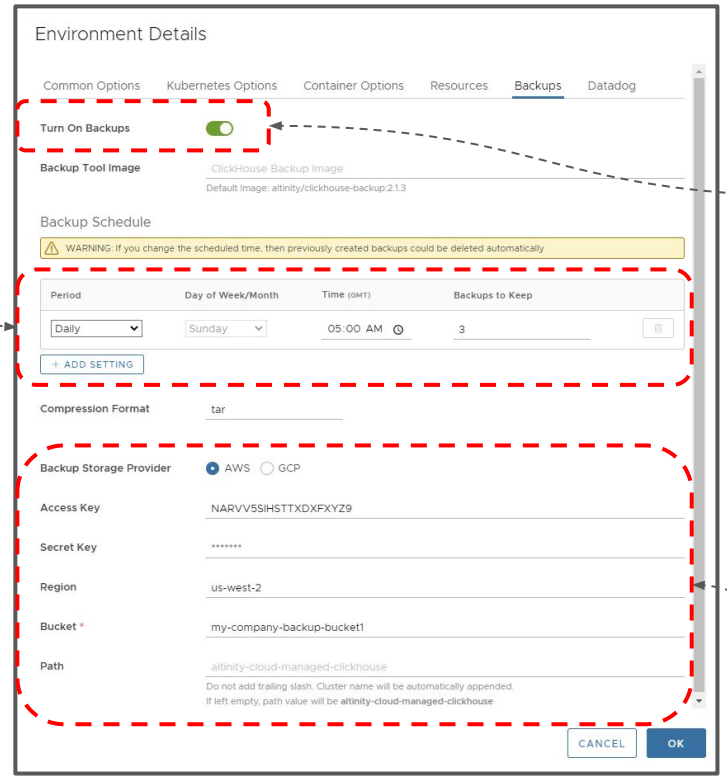
Rescaling is limited by your resources



Add new node types from the Environment Dashboard



Configure backups in Environment->Edit->Backups

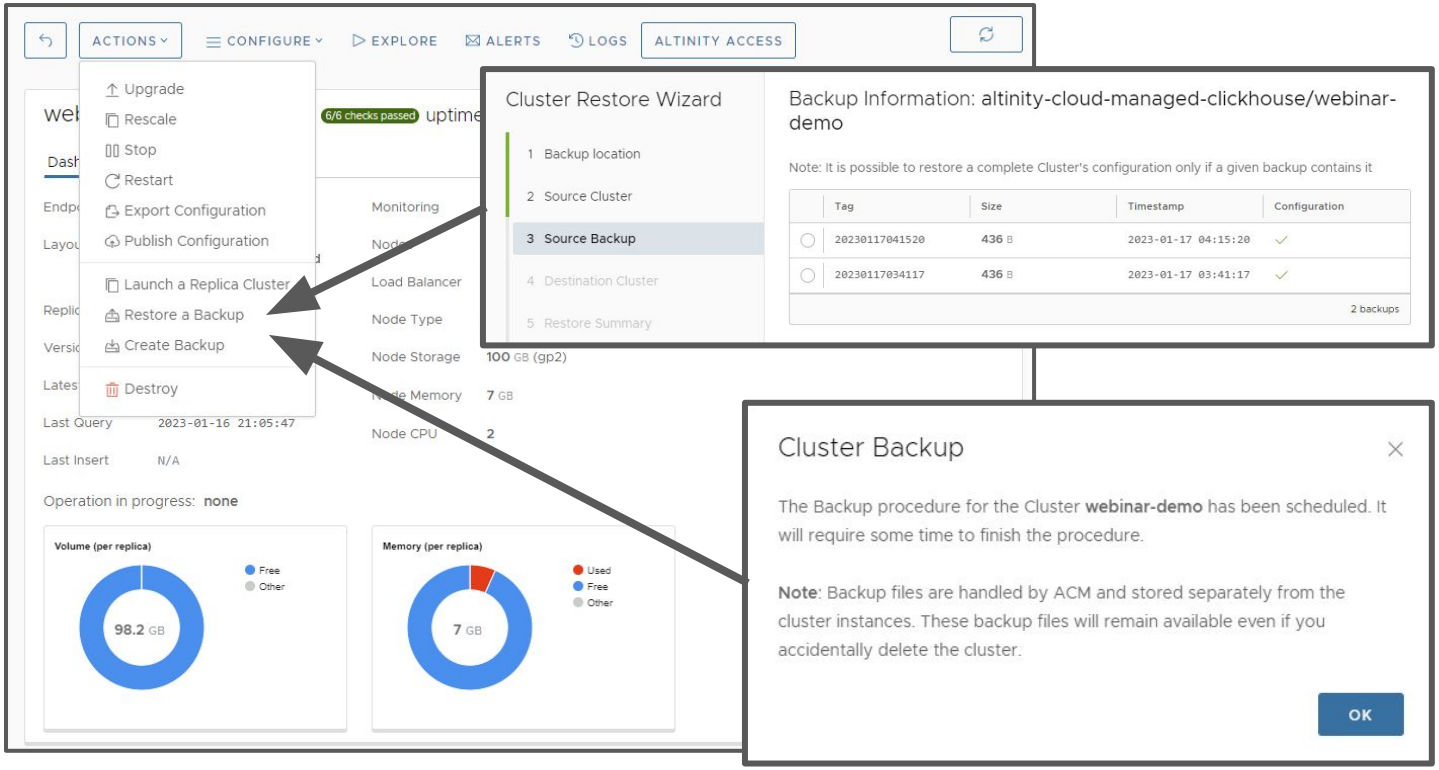


Backup schedule

Enable backups

S3 bucket details

You can backup and restore Kubernetes data



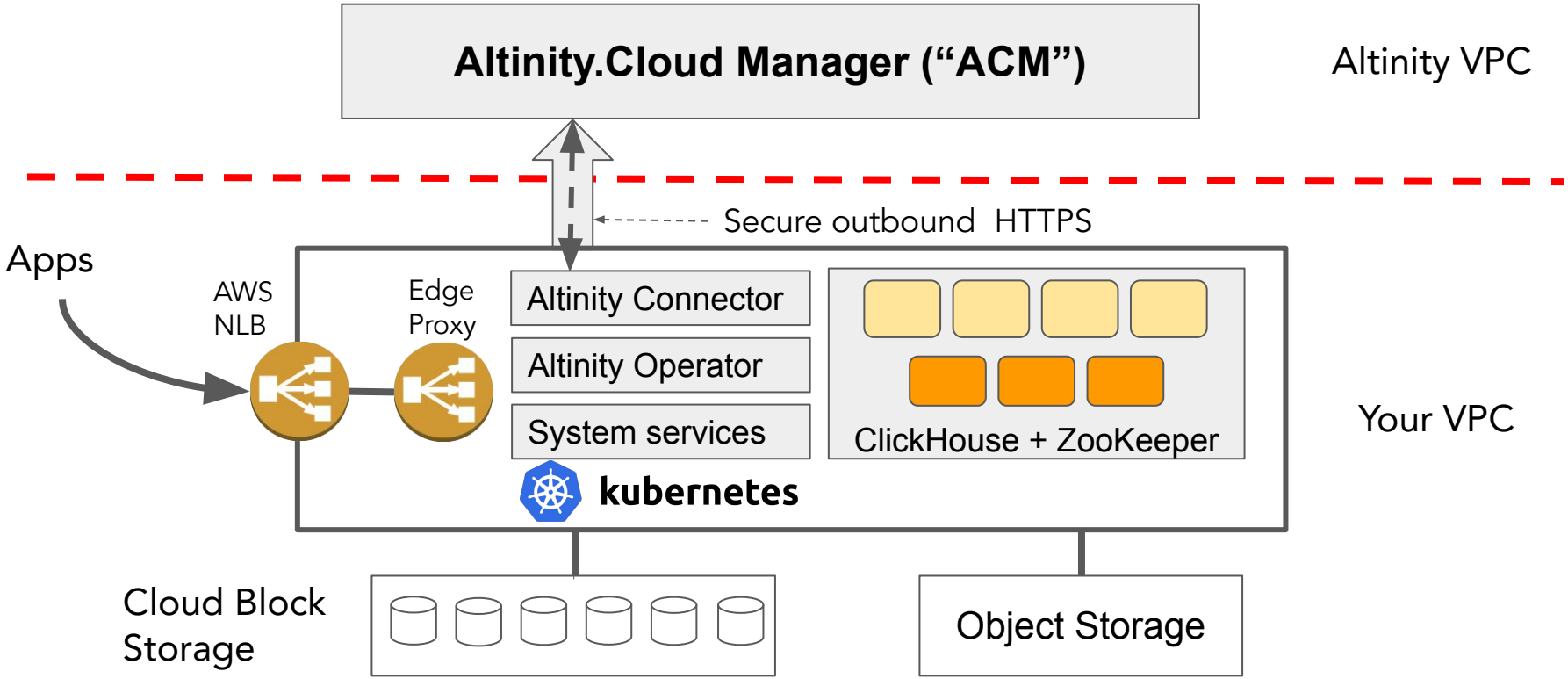
What else is there?

- Setting up private connectivity from Kubernetes to other VPCs
  - ClickHouse.Cloud supports VPC Endpoints
- Running other applications alongside managed ClickHouse
- Disaster recovery

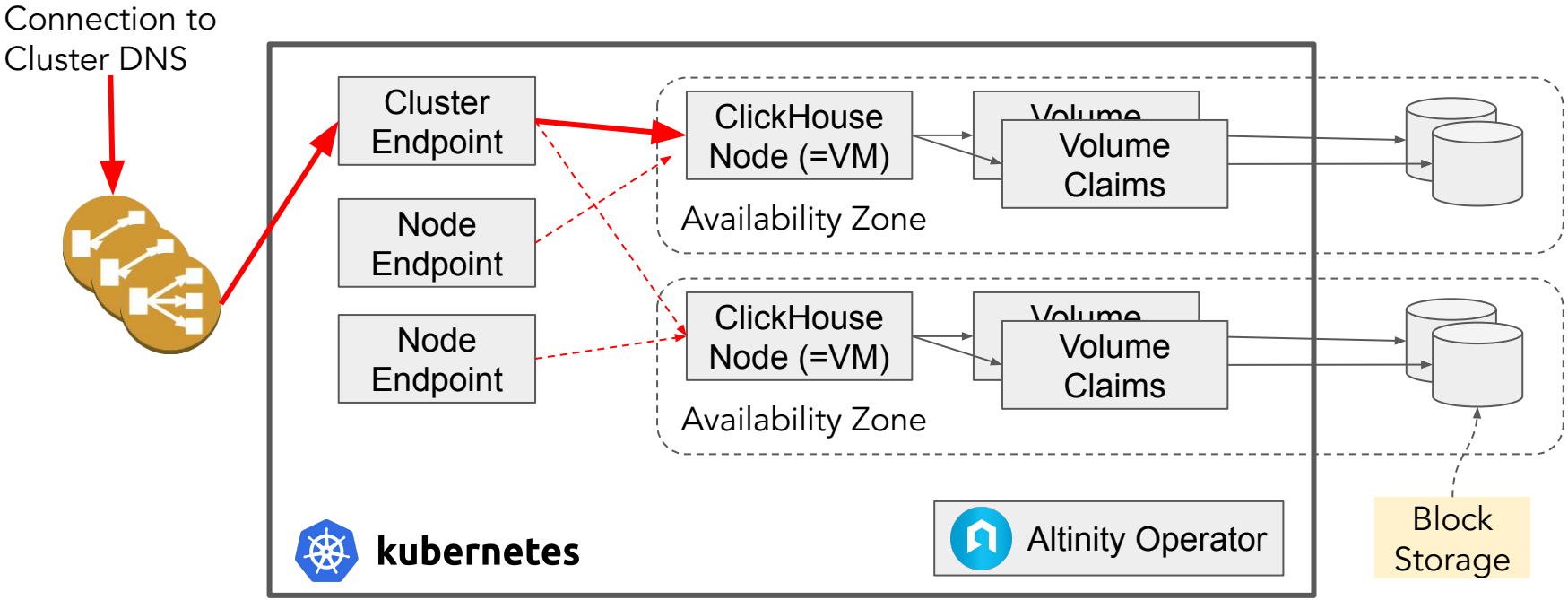
Contact support for help on any of these!

Advanced Topics

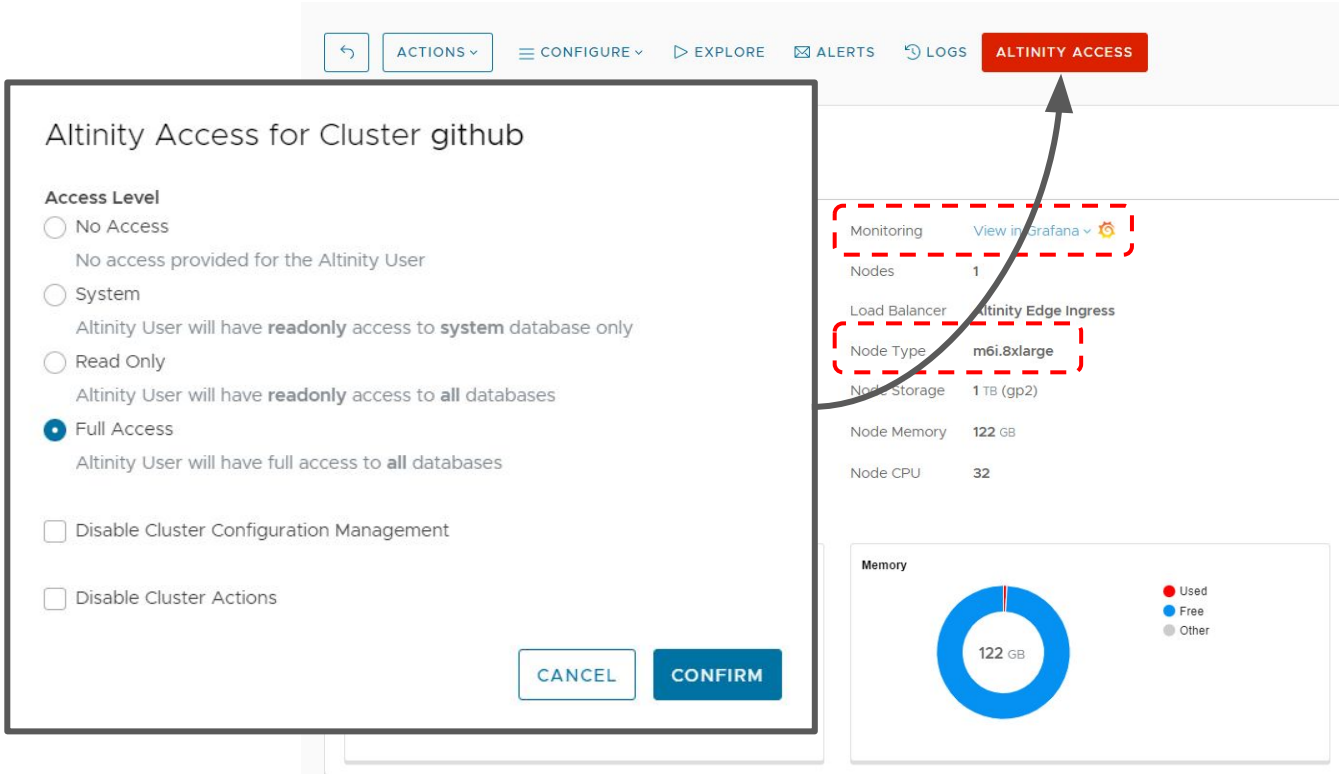
How does Altinity.Cloud Anywhere work internally?



What's going on under the covers in a ClickHouse cluster?



Restricting access to data is a standard feature



How Altinity.Cloud Anywhere support works

Altinity.Cloud has enterprise support for all environments

- Schema design
- Performance optimization
- Troubleshooting
- Upgrades
- Capacity planning
- Integrations (Kafka, BI, librarise.)
- And random questions

We also answer questions about Kubernetes!

Ways to access support

- Slack - shared slack channel
- Zendesk - email to support at altinity dot com

How to get the most out of Altinity.Cloud support

- Log production issues in Zendesk, not Slack
  - Makes it easier to track progress
- Contact us in advance prior to:
  - ClickHouse version upgrades
  - Scale-out operations that may require us to raise AWS or GCP quotas
- Don’t wait for problems. We can help you with migration, design and capacity planning and many other topics
  - Just ask if you have questions!
  - We also can do regular check-in calls to help you make progress

Wrap-up

- Altinity.Cloud Anywhere gives you convenience of cloud with the control of installed software
- Pricing is based on compute only. We don't charge for data
- Run ClickHouse anywhere that Kubernetes runs
- Don't want to bother with Kubernetes? We can install that, too!

Start a free two-week trial today!
https://altinity.com/altinity-cloud-anywhere/

Thank you!
Questions?

Website: https://altinity.com
Email: info@altinity.com
Slack: altinitydbworkspace.slack.com

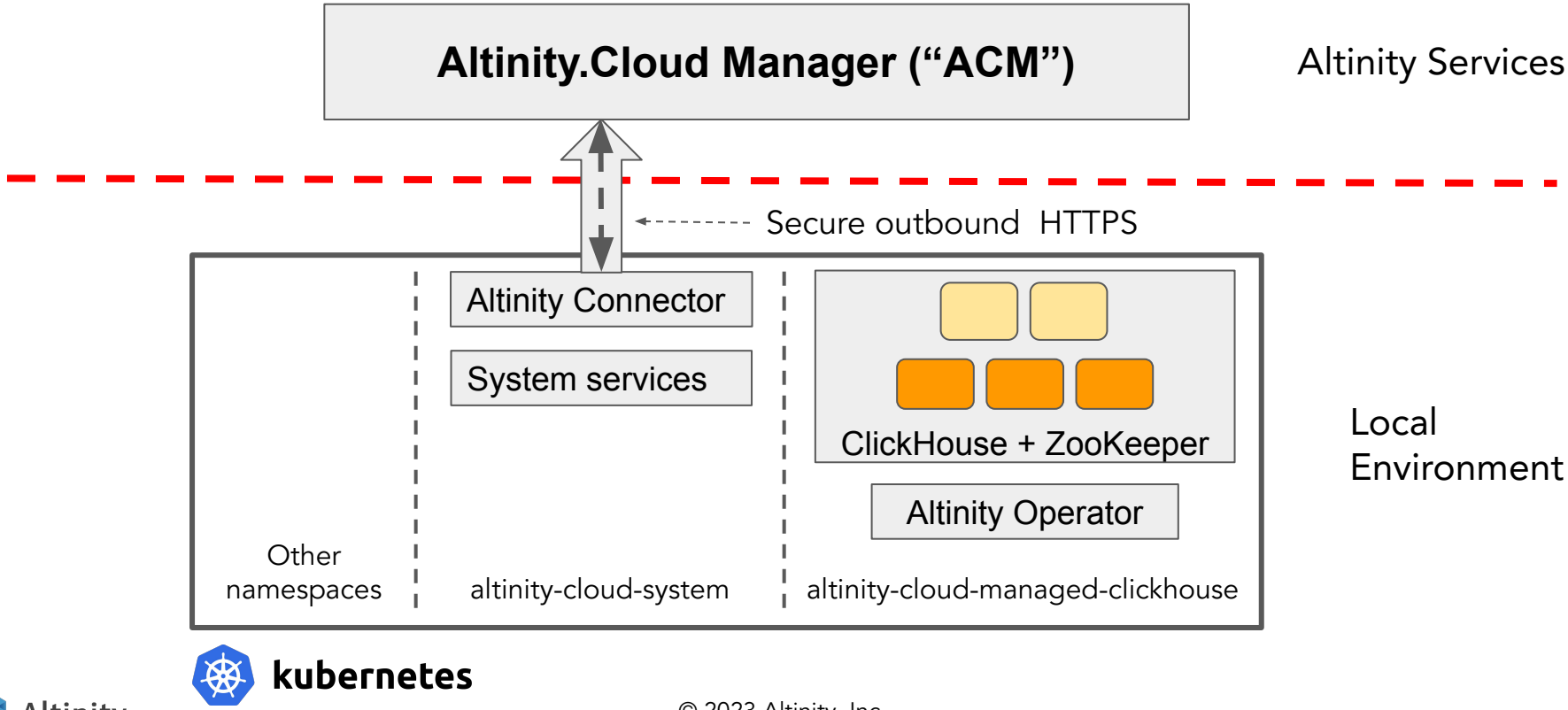
Altinity.Cloud

Altinity Support

Altinity Stable Builds

Free 2 week trials

How does Altinity.Cloud Anywhere work internally?



Altinity.Cloud Anywhere can run in your Kubernetes